CMPSC 111
Introduction to Computer Science I
Fall 2016

Practical 2
September 9, 2016
Due in Bitbucket by midnight of the day of your practical
"Checkmark" grade

Summary

Create a Java program that prints something "interesting", and then use the `git add`, `git commit`, and `git push` commands to upload it to your Git repository hosted by Bitbucket. See the end of the assignment for a few hints and suggestions for creating Java programs that perform output.

Review the Textbook

In addition to studying the text of and slides for Chapter 1, be sure to quickly read Section 2.1 of your book as it explains how to print some of the special sequences, a topic that is also discussed at the end of this assignment; in particular, please examine the examples of escape sequences in Figure 2.1. Please see the course instructor if you have questions about using escape sequences.

Exercise: Print Something "Interesting"

Create a Java program that uses a sequence of no more than ten `System.out.println` statements to print something "interesting." You may not use any other features of Java, such as variables, loops, etc. However, you are required to use at least one of the "escaped" sequences, such as `\"` or `\\`. Remember, it is possible to get pictures that are taller than ten lines by using the `\n` sequence in your `println` statements. Please see the instructor if you have questions about this requirement.

Your program must print your name and today's date (using "`new Date()`" in a `println` statement). This will not count as part of the ten required print statements.

You must come up with an original design—under no circumstances should you directly copy a design from another source, such as an "ASCII Art" web site. (However, you may look at such sites for inspiration.) The next two pages furnish two examples that you are encouraged to try. As you complete these two examples, please take time to pause and reflect on why the program produces the output that it does. If you cannot get one of these examples to work, then please show a teaching assistant or the course instructor the problem that you are encountering. Remember, when you are programming it is important for you to work incrementally by typing a few lines of code and then trying to compile your program and resolve any errors surfaced by the compiler.

Preparing for the Programming Task

Before you start creating the Java program required by this assignment, you should separately type the commands "`cd cs111S2016`", "`cd cs111S2016-<your user name>`", and "`cd practicals`" in your terminal window. Once you are in the `practicals/` directory of your Git repository, you can

type the command "`mkdir practical02`" to create a new directory for this assignment. You can run the `gvim` command from this directory when you are ready to begin to implement the required Java program. Please see the instructor if you have problems with these preparatory steps.

Example 1: File "`PrintName.java`"

```
//*********************************
// Bob Roos
// Practical 2, 11-12 September 2015
//
// Prints a name 'Bob'
//*********************************
import java.util.Date;
public class PrintName
{
  public static void main(String[] args)
  {
     System.out.println("Bob Roos, CMPSC 111\n" + new Date() + "\n");
     System.out.println(" ____         __");
     System.out.println("  |  \\         |");
     System.out.println("  |__/   __    |__");
     System.out.println("  |  \\  /  \\  |  \\");
     System.out.println(" _|__/  \\__/ _|__/");
  }
}
```

OUTPUT:

```
javac PrintName.java
java PrintName
Bob Roos, CMPSC 111
Wed Sep 10 21:04:41 EST 2015


 ____         __
  |  \         |
  |__/   __    |__
  |  \  /  \  |  \
 _|__/  \__/ _|__/
```

Example 2: File "`PrintFace.java`"

```
//*********************************
// Janyl Jumadinova
// Practical 2, 11-12 September 2015
// Prints a face
//*********************************
import java.util.Date;
public class PrintFace
{
  public static void main(String[] args)
  {
     System.out.println("Janyl Jumadinova, CMPSC 111\n" + new Date() + "\n");
     System.out.println("   \\\\\\\|||///");
     System.out.println("   /       \\");
     System.out.println("   | -- -- |");
     System.out.println("  @|  O O  |@");
     System.out.println("   |   V   |");
     System.out.println("    \\ \\_/ /");
     System.out.println("     \\___/");
  }
}
```

OUTPUT:

```
javac PrintFace.java
java PrintFace
Janyl Jumadinova, CMPSC 111
Wed Sep 10 21:11:55 EST 2015

   \\\|||///
   /       \
   | -- -- |
  @|  O O  |@
   |   V   |
    \ \_/ /
     \___/
```

Correctly Using Version Control

As you are typing your program in the `gvim` text editor, you should regularly save your files. Once you have created a preliminary version of your program and it compiles and runs as anticipated, you should use the "`git add`" command to "stage" it in your Git repository. Next, you can use the "`git commit`" command to save it in your local repository with a version control message. Finally, you can run "`git push`" to transfer your file to the Bitbucket servers. For this practical assignment,

you do *not* have to hand in a hard copy of anything—just upload your Java program and an output produced by your program to Bitbucket using the appropriate `git` commands. Please review your "Git Cheatsheet" and talk with a member of the class, the course instructor, or a teaching assistant if you do not understand how to use some aspect of the Git version control system.

Hints About Java Programming and Escape Sequences

The name of your program file (for instance, "`PrintFace.java`") must be the same as the name in the "`public class ...`" statement—see earlier examples where you practiced this skill.

The sequences "\" (backslash) and "`"`" (double-quote) require special handling. To print them out, you need to put an extra "\" in front of them. For instance,

The statement:	`System.out.println("backslash: \\, quote: \"");`
prints:	`backslash: \, quote: "`

Since this is one of our first practical assignments and you are still learning how to use the Java programming language, don't become frustrated if you make a mistake. Instead, use your mistakes as an opportunity for learning both about the necessary technology and the background and expertise of the other students in the class, the teaching assistants, and the instructor.

General Guidelines for Practical Sessions

- **Experiment!** Practical sessions are for learning by doing without the pressure of grades or "right/wrong" answers. So try things! The best way to learn is by intelligently experimenting.
- **Submit *Something*.** Your grade for this assignment is a "checkmark" indicating whether you did or did not complete the work and submit something to the Bitbucket repository using the "`git add`", "`git commit`", and "`git push`" commands.
- **Practice Key Laboratory Skills.** As you are completing this assignment, practice using the `gvim` text editor and the Ubuntu terminal until you can easily use their most important features. Additionally, ask a teaching assistant or the course instructor to teach you some of the advanced features of `gvim` and the terminal, thereby helping you to work more effectively.
- **Try to Finish During the Class Session.** Practical exercises are not intended to be the equal of the laboratory assignments. If you are simply a slow typist, I've given you until the end of the day, but ideally you should upload a file, even a partially working one, by the end of the class period. So, make sure that you correctly upload your file to your Git repository!
- **Help One Another!** If your neighbor is struggling and you know what to do, offer your help. Don't "do the work" for them, but advise them on what to type or how to handle things. If you are stuck on a part of this practical session and you could not find any insights in either your textbook or online sources, formulate an intelligent question to ask your neighbor, a teaching assistant, or a course instructor. Try to strike the right balance between asking for help when you cannot solve a problem and working independently to find a solution.
- **Update Your Repository Often!** You should `add`, `commit`, and `push` your updated files each time you work on them, always including descriptive messages about each code change.
- **Review the Honor Code Policy on the Syllabus.** Remember that while you may discuss programs with other students in the course, programs that are nearly identical to, or merely variations on, the work of others will be taken as evidence of violating the Honor Code.